A Temporal Pattern Predictor for Virtual Characters

Micah Rosenkind[1], Prof. David Arnold[1], Dr. Graham Winstanley[1]

[1]University of Brighton, UK
m.m.rosenkind@brighton.ac.uk

Abstract

This paper discusses a prototype of a temporal pattern predictor, which was built on specifications derived from the descriptions of the "Ergotrix" temporal memory network in Valentino Braitenberg's "Vehicles" (Braitenberg, 1984). The prototype was developed as a component for a control architecture for virtual characters.

Introduction

In Valentino Braitenberg's "Vehicles - Experiments in Synthetic Psychology" (Braitenberg, 1984), the author describes a temporal pattern memory component that enables the synthetic creatures to base their behaviour on past experiences: Rather than just reacting to the currently perceived sensory stimuli, these agents first form predictions based on the statistical probability of previously perceived patterns re-occurring. These predictions then form the basis for their actions. In other words, the creature reacts to a pattern of expected stimuli, instead of waiting for the stimulus to actually occur.
Braitenberg's temporal pattern memory component is implemented in the form of a connectionist network of nodes. In such a network, each node may represent the presence of a sensory stimulus, a pixel on a screen or a sensory 'value' of another kind (e.g. the distance value of an infra-red sensor). The individual nodes can potentially form connections (which are often referred to as associations) to any other nodes in the network. The strength of the connection is determined by the connection weight.

In many classical neural network models, such as feed-forward or back propagation networks, (e.g. McCulloch Pitts 1943, Bryson and Ho 1969,), the connection weights are set using sophisticated learning algorithms. However, many of these separate the process of setting weights (learning or training phase), from actually using the network (execution phase). The model presented in this paper, uses a learning model that can form new associations while it is being used. This learning rule is based on Hebb's "what fires together, wires together" postulate (non-verbatim from Hebb, 1949), which itself tried to summarize the general behaviour of connecting neurons in biological brains.
Regardless of the learning rule used, traditional implementations of connectionist networks will see that a signal passes from active nodes to all their connected peers. The amount of activity transmitted is determined by the weight of the connection.

In addition to the properties that define a Hebbian learning-based neural network, Braitenberg specifies two further key characteristics, which our network model intends to address:

1. The memory network associates only elements that are active in succession, within a brief delay and not those, which are active simultaneously. This differentiates it from a basic associative connectionist network.
2. Memorized patterns can be reproduced at an arbitrary speed. If they are reproduced at a more rapid pace than they are likely to occur as sensed via the sensory system, the network acts as a predictor.

These two functional requirements were at the core of a series of our incremental prototypical experiments.

Methods

A Fixed-Delay Network

After initial investigation into delay-line networks, which used timers to record variable activation delays between a set of elements (see Figure 1), we employed a network with fixed delays between the activation of elements. The previous models had focused on implementing the ability of modelling a variety of time delays between device pairs in order to learn temporal patterns in threshold device populations. The model discussed in this paper moves away from this variable delay paradigm towards a notion inspired by the way film cameras record the passage of time. Film cameras are usually set to operate at a fixed frame-rate; say 25 frames per second (fps). When a slowly moving object passes by the camera lens it leaves a contiguous trace of strongly activated pixels on the film (or CCD chip for that matter). On the other hand, if a fast moving object passes by the camera lens, it would only leave trace signals on a few, further apart areas of the film/CCD chip. Thus the speed of the object is represented by two distinct patterns on the film, even though the delay between the activation of the neighbouring contiguous pixels was the same in both cases. The time delay between the contiguous and distant impressions was 1/25 of a second.
This model uses the idea that speed is not a matter of delaying information transmission between devices. At one time-step the delay between one node activating another is always

constant. Rather, time is a property of an observed object and the pattern that its observation leaves in the connections of a perceiving network. Figure 1 illustrates the fixed delay network. The activation timing between each pair of devices in the network is fixed. The timings are not stored in the connections, but in the patterns that are left when sensed by a group of connected devices.

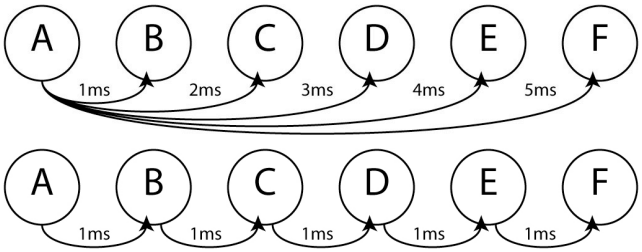

Figure 1 Illustrates variable delay and fixed delay paradigms

Applying the above example to a network of connected nodes ABCDEF, that are aligned in the visual field of a camera from left to right and an object passes by them at slow speed, then all devices A-B-C-D-E-F will be activated in sequence. If the object passes by at high speed, some visual receptors might actually fully activate- in this case only triggering the sequence A-C-E for example.

Thus, even though the activation speed between the threshold devices in the visual receptor is constant, they can capture different timings/speeds within varying received activation patterns.

Comparing this theoretical model to the subjective experience of observing moving objects, an object moving past a photo-cell faster would allow less light to pass from it into any given receptor - therefore triggering a weaker activation signal. In addition there even seems to be a threshold for the maximum speed that can be perceived by a given visual system. A very high velocity object, such as a spinning airplane propeller may appear to be entirely transparent, with only a slight "dimming" of the background signifying their presence.

To summarize: If time is a property of the object being observed and is not represented in the network as a property of the connections between nodes, then time is perceived and encoded as a pattern in the network and not as a value. This makes time a relation between an observed object's speed/rate of movement and an observer's perception/processing rate.

Based on initial observations, the predictor model had to address the following issues:

1. Feedback loops can create uncontrolled activity and irrecoverable states.
2. One time-step is not sufficient for accurate long-term predictions. A short-term, working memory should be used to extend the predictor.
3. Currently, every association is stored, leading to a quick saturation of the network. Competition and pattern decay should be introduced to:
    a. Resolve conflict between opposing patterns
    b. Decay old and rare patterns over time

Results

The Algorithm

The first version of a model based around the notion of a fixed-delay network used a very simple algorithm. At each time step, the currently firing network node is associated with the node that fired on the previous time step. The association is uni-directional, meaning that only the connection *from* the previous node is reinforced, while the connection *to* that node is not reinforced. The association weight is determined by the time that has passed since the previous node fired.

The initial implementation used a counter to determine the time that had passed between the firing of the two connected nodes. However, due to requiring counters for each node that fires during a single time step, the revised implementations instead use an extension of the action-potential charge of each node. While traditional artificial neural networks use a binary charge state of either 1 or 0, this extended model still outputs a binary energy value, but instead of returning to 0 immediately, the value is gradually decreased over a series of time steps. This made it possible to use the *charge falloff* as an individual measure of time for each node.

Below is the pseudo code for the algorithm used:

```
For each node A in the network

  //Update CURRENT and PREDICTED charge:
  •Retrieve charge from previous time step
  •Decay the previous charge
  •Add external stimuli to charge

  •Calculate internal stimuli
    •For each node B (that is not A) that
    fired on the previous time step (in
    fired list)
      •If previous node B is connected to
      current node A
        •Calculate and add the input
        charge from B to A (node B's
        output & weight from B to A)
        •Decay the connection weight from
        A to B (if a connection exists)

    •Update current charge for node A
    (previous charge + external stimuli)
    •Update the predicted charge for node
    A (internal stimuli / past effect
    scaling value)

  //Fire Nodes and Update Associations:
  •If current charge > threshold
    •For each node B (that is not A) that
    fired on the previous time step
      •Increase the weight from B to A
      •Decrease the weight from A to B (if
      a connection exists)
    •Add the current node A to the fired
    list
    •Remove the oldest element from the
    fired list
```

The model showed the capability of reproducing time series patterns, albeit without any intermittent pauses between node activations. The pattern *sequence* was reproduced faithfully, but the reproduction was sped up with each node activating exactly one time-step after the previous one.
Another issue was that pattern loops could end up creating feedback patterns similar to Conway's "Game of Life" (Gardner, 1970).

Separating Internal Activity from Prediction

The first change from the previous model was that the *predicted activity* was separated from the *actual activity* in the network. Instead of adding the internal activation energy to the total *charge* of each node (which determines whether the node would fire), it is instead accumulated in a new node property, the *prediction*. In the visualization that was used in our simulation, the *prediction* and the *charge* are displayed as two separate coloured bars to make it clear to the observer which was the *charge* caused by external stimulus and which was the predicted path. Separating the two immediately resolved the internal feedback problem of course. Figure 2 shows the implementation of the temporal pattern predictor which visualizes the separation between sensory input and prediction. The left most active threshold device is currently active. The two paths emanating from it are two previously perceived patterns of node activity. The darker shading indicates that a predicted pattern has been perceived more often and/or more recently than the weaker prediction.

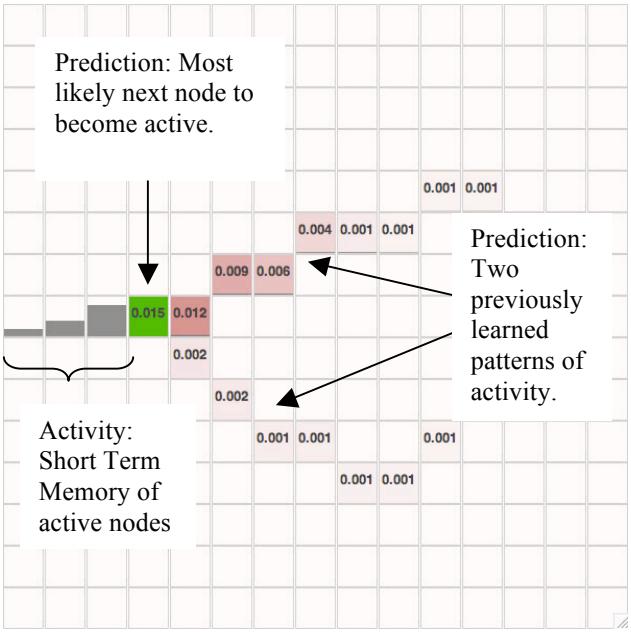

Figure 2 The Predictor model implementation

Short Term Memory Trajectory

The main problem that occurred with the previous version of the predictor was that of pattern interference. In a network, any given node may be part of several trained patterns. When this node is activated, how does the network *choose* which pattern to activate? While any of the patterns are valid when taking into account only the current state of the network, viewed as a series of states in *time*, the idea of all being valid becomes less likely. To illustrate this, the diagram in Figure 3 shows an experiment that sees four different opposing stimulus sequences presented to the network. Each of the four sequences passes through the same middle node and sequence pairs 1&2 and 3&4 share the same path. Figure 4 shows the false predictions the current model makes.

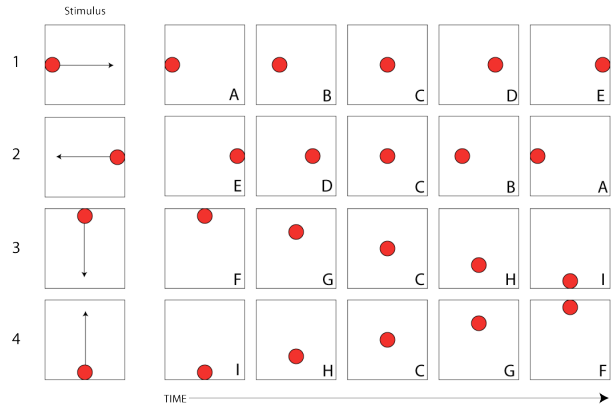

Figure 3 Experiment setup to test pattern interference

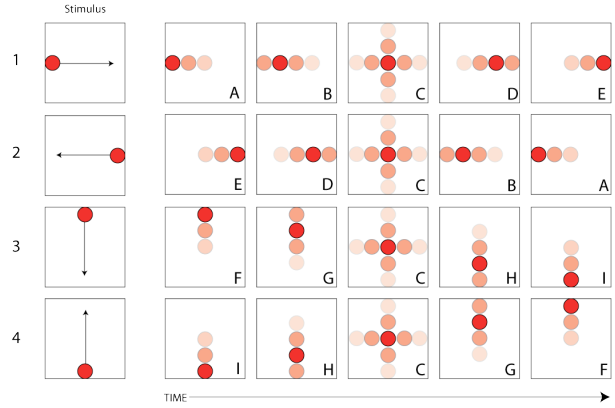

Figure 4 The false predictions the network makes due to pattern interference

Note how only 9 unique states can be identified from the perspective of this network. Viewing state C without taking the preceding time-steps into account, the predictor could validly predict either state D,B,H or G as the next possible position in the sequence.
In order to deal with this problem of differentiating between different temporal patterns, a short-term memory was introduced into the model. This sees a series of *past devices* associated to the current device under the notion that they are precursors of the currently active device. The further in the past these devices are active, the weaker the association to the current device. Figure 5 is an illustration of this mechanism. It shows an example of impact of short-term memory the possible predicted paths A, B and C. While all three are equally likely from the perspective of the current node, the

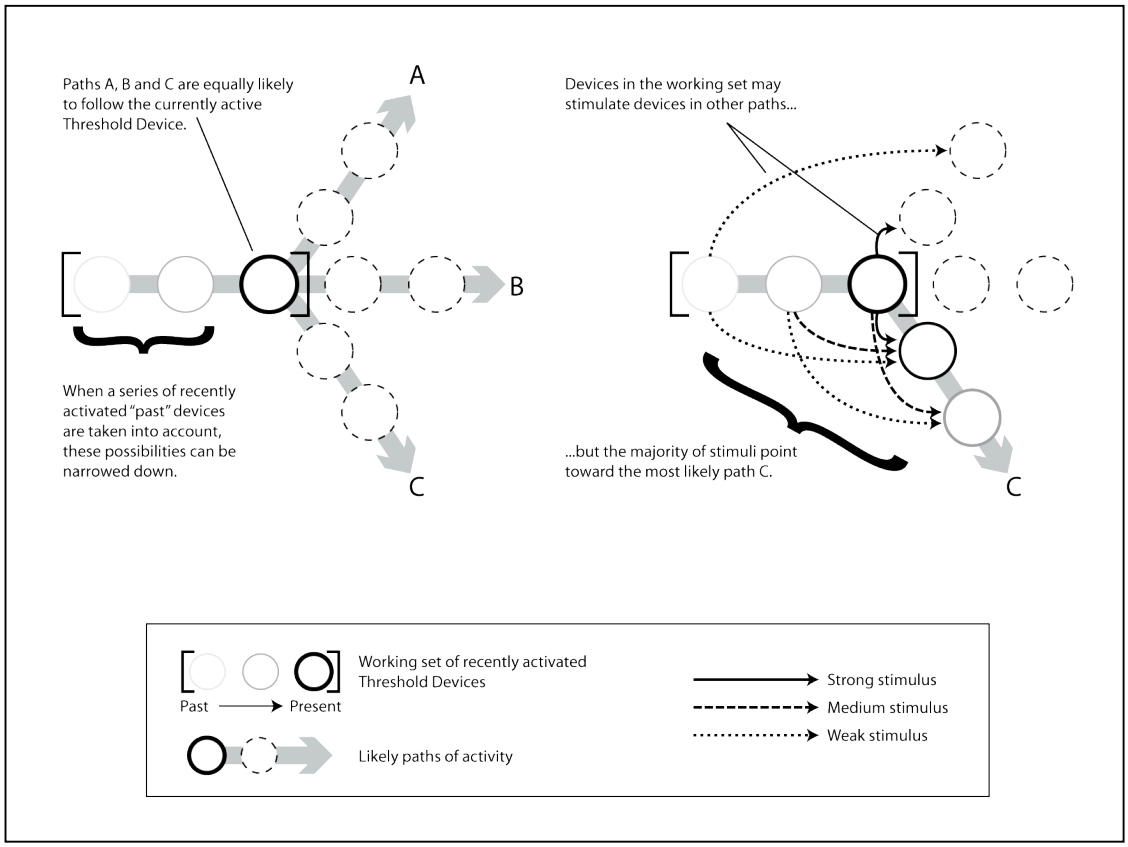

Figure 5 Diagram illustrating how the short-term memory allows the network to differentiate between predicted patterns.

additional stimulus from the two previous nodes in the short-term memory accumulate to support path C as the most likely pattern.

The result of adding a short-term memory of past nodes to the model is illustrated in Figure 6. The *perception* of the predictor has changed as it keeps the series of past nodes in mind. From the outset, this allows the network to differentiate between 20 unique states instead of just 9.

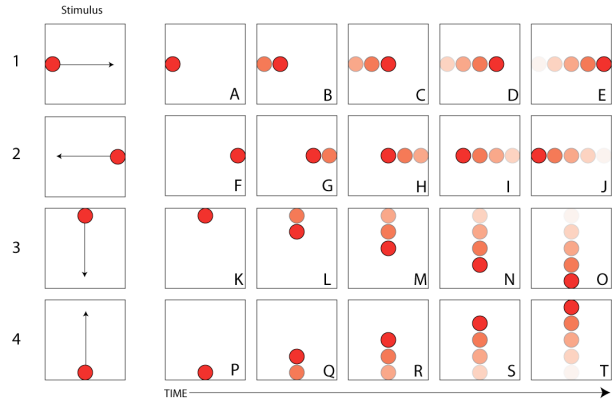

Figure 6 Introducing a short-term memory of active nodes changes the perception of the network

Based on the altered perception, Figure 7 shows that these 20 states are then associated with 20 different predictions. Since the influence from the short-term memory adds additional activity to the network, the problem of saturating the entire network becomes relevant. To deal with this problem the model needs to include the notion that certain patterns are *competitive* in that they represent opposing positions to a fact.

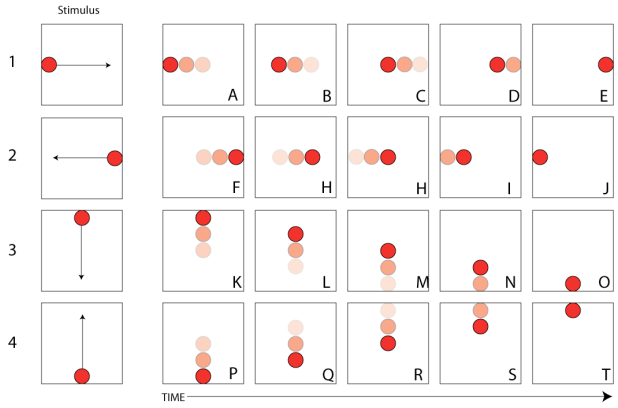

Figure 7 Using a short-term memory allows the network to differentiate between predictions

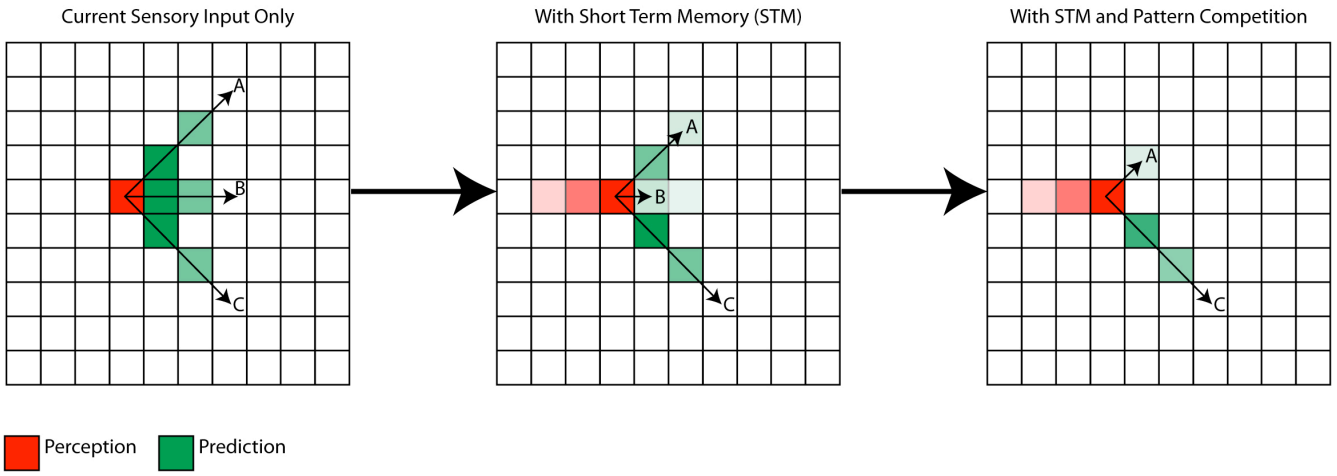

Figure 8 Pattern Inhibition enhances the prediction by inhibiting patterns only pointed to by individual nodes.

Pattern Inhibition

A simple form of competition that was tested with this model was inhibition between patterns. Inspired by the lateral inhibition algorithm commonly used in edge-enhancement algorithms in computer vision, this mechanism allows currently active nodes and the nodes in short-term memory to inhibit all the nodes that they are currently not connected to. The result is a cumulative effect of inhibition. In parallel with exciting connected nodes as depicted in Figure 5, past and present nodes will inhibit every other node in the network, thus *enhancing* the strongest mutual patterns among them. Figure 8 illustrates this effect.

Discussion and Further work

Since the model uses fixed delays, the predicted sequence is triggered simultaneously. Speeding up a predicted pattern is certainly desirable according to Braitenberg, who states that a predictive brain would need to "reproduce sequences at a more rapid pace" (Braitenberg, 1984, pg. 72). However, the interval information between the activation of nodes is lost, which contradicts Braitenberg's earlier statement that "we implicitly assumed that the Ergotrix wires would be trained to reproduce sequences of activation at the same pace as the original occurrence of the sequences of events." (Braitenberg, 1984, pg. 72).

Overcoming this issue would require re-introducing some form of internal self-activation to the network, albeit with a mechanism for preventing undesirable states such as feedback loops.

While this internal dynamic is desirable in a future model of the predictor mechanism, a feedback-free predictor has the benefit of stability and the ability to clearly visualize the discrepancy between sensed input and predicted output. It might also be possible to derive the timings from the prediction value of nodes.

Further Work

Testing the predictor in an embedded scenario is the primary priority at this point. Two experiments are currently being prepared. The following is a summary of some of the early findings.

The scenario for both experiments is a chase-and-catch setup, with the goal being to stay as close as possible to a moving target. The simulation features a differential drive-driven driven agent with two distance sensors, which can be set to track a target. The aim of these experiments is to see whether adding the ability to anticipate the path of a moving target leads to more optimal behaviour. Note that in both experiments, the tracked target is moving *faster* than the agent. To catch the target, the agent therefore needs to intercept the target:

The first experiment implements the predictor as a probabilistic occupancy map (POM), inspired by previous work by Damian Isla (2002a, 2002b). The experiment sees the predictor network represent the location of a tracked object on an occupancy grid. The predictions generated therefore propose the possible future location of the tracked object and an agent's sensors can be set to track the prediction instead of the 'actual' tracked object. This can be implemented using the current predictor, which only tracks a single point (the location of the object on the map).

Figure 9 shows the path of the agent without using the predicted position of the target, indicated by a cross. Figure 10 shows that the agent manages to get closer to the target when using the predicted position as the input. While this early result shows that our predictor can function as a POM, the result is still highly dependent on tuning. Further Increasing the speed of the target, would require an agent that is capable of adopting a strategy that does not involve it following the target, but instead waiting for it at an expected location. The current predictor would need to be extended to allow for such behaviour.

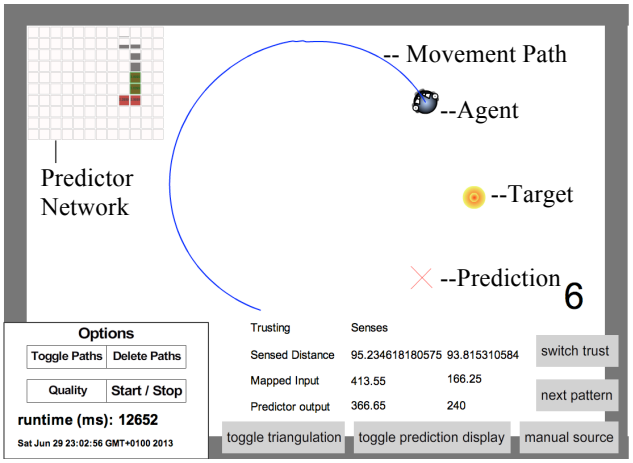

Figure 9 Chase-scenario with a moving target. The objective is to stay as close to the moving target as possible.

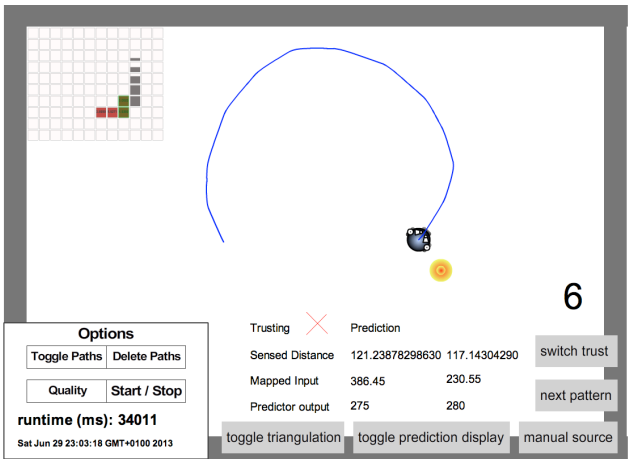

Figure 10 Using the predicted position of the target improves the behaviour.

Figure 11 shows the second experiment, which uses the predictor to generate a multi-dimensional sensory space. By using multiple individual predictors, one for each sensor input, separate predictions of the expected future state of each sensor are generated. In this case, the sensors used are two distance sensors. To function with the current predictor model, the distance reading need to be converted into discrete values that can be mapped to the predictor grid. Thus finding a suitable approach to balancing the performance and the accuracy of the predictions will be of particular interest.

The two distance sensor readings are not sufficient to accurately triangulate and predict the position of an object's position, since the distance sensors are omni-directional and always return a positive distance measurement in our simulation. The agent therefore requires an additional internal calculation that tells it in which direction (in front or behind) the observed object is. Combining this with the sensor readings thus gives discrete values ranging from negative

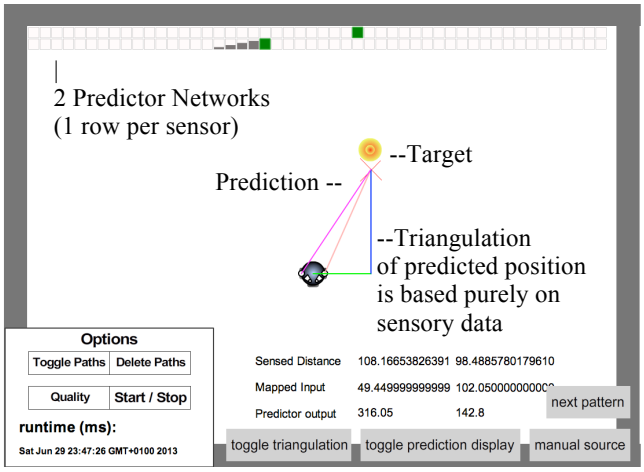

Figure 11 Experiment testing the use of the predictor network to anticipate sensory data. The future position of the target is triangulated and displayed.

(behind) to positive (in front) that can be fed into the predictor network. While this model already works for a stationary agent, the main issue that remains is effect of self-movement on the sensed values. This feedback-loop between the relative position of the agent with regard to the tracked object has to take into account, or rather counteract, the effect that self-movement has on the sensory data. Without this additional system in place, the current prototype will only work in a stationary agent, observing and predicting the location of a moving tracked object using its distance sensors.

Conclusions

Our model of a Braitenberg-inspired temporal pattern predictor can successfully predict and visualize the path of a moving object and can avoid interference between crossing patterns through the use of short-term memory. A set of experiments successfully tested the model in the context of a chase scenario and has revealed several ways in which the current model could be further improved.

In the first experiment, the choice between following the sensed position of the target versus the predicted position on the occupancy map was controlled by the experiment. Using the user interface we were able to switch between the two and compare the resulting behaviours of the agent. A central topic for further research on our particular predictor model is the inclusion of an automatic switching mechanism that enables the agent to make a choice between purely reactive behaviour (following the sensory input directly) and pro-active behaviour (following the internal representation of the target on the POM). This in turn could be extended to include the ability to optimise the prediction mechanism by re-enforcing accurate predictions. Braitenberg's original description of the predictor includes this functionality and suggests a method of positive reinforcement based on classical conditioning (Pavlov, 1903).

The current model only supports sequences of single nodes. It only allows a single *past node* to be associate with a single

current node. To allow for more complex patterns and a wider range of application, it should be possible to associate groups of *past nodes* with groups of *current nodes*. The ability to support multi-point patterns could improve the predictor further. Connecting the currently separate sensor readings to the same network should give the predictor more evidence to base individual predictions on. As we saw with the inclusion of short-term memory, this could potentially improve its ability to differentiate between similar patterns.

Acknowledgments. This work was supported by the University of Brighton (UoB) Cultural Informatics Research Group, the UoB School of Computing, Engineering and Mathematics and the UoB Doctoral College Research Student Conference Support Grant.

References

Braitenberg, V. (1984). *Vehicles: Experiments in Synthetic Psychology*, The MIT Press. Cambridge, MA, USA

Bryson, A. E. Ho, Yu-Chi (1969). *Applied optimal control: optimization, estimation, and control*. Blaisdell Publishing Company or Xerox College Publishing. p. 481.

Gardner, Martin (1970-10). *Mathematical Games - The fantastic combinations of John Conway's new solitaire game "life"*. Scientific American 223. pp. 120–123. ISBN 0-89454-001-7.

Hebb, D. O. (1949). *The Organization of Behavior*, Wiley and Son.

Isla, D. and Blumberg, B., (2002a). *Object Persistence for Synthetic Creatures*, Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS): Part 3, ACM, New York, NY, USA: 1356-1363

Isla, D. and Blumberg, B., (2002b). *New challenges for character-based ai for games*

McCulloch, W. and Pitts, W. (1943). *A logical calculus of the ideas immanent in nervous activity*. Bulletin of Mathematical Biology 5(4): 115-133

Pavlov, I. P. (1903) *The Experimental Psychology and Psychopathology of Animals From Nobel Lectures*, Physiology or Medicine 1901-1921, Elsevier Publishing Company, Amsterdam, 1967